Apache Hadoop Installation and Cluster setup on AWS EC2 (Ubuntu) – Part 2

A guide to install and setup Multi-Node Apache Hadoop Cluster on AWS EC2

edureka!

9/20/2013

APACHE HADOOP INSTALLATION AND CLUSTER SETUP ON AWS EC2 (UBUNTU) – PART 2

A guide to install and setup Multi-Node Apache Hadoop Cluster on AWS EC2

Table of Contents

Introduction

This setup and configuration document is a guide to setup a Multi-Node Apache Hadoop cluster on Amazon Web Services (AWS) Elastic Cloud 2 (EC2) using ***'free tier usage eligible'*** Ubuntu (t1.micro) servers. If you are new to both AWS and Hadoop, this guide comes handy to quickly setup a Multi-Node Apache Hadoop Cluster on AWS EC2.

Note
AWS also provides a hosted solution for Hadoop, named Amazon Elastic Map Reduce (EMR) but Only Pig and Hive are available as of now and with a cost.

The guide describes the whole process in two parts:

Section 1: Setting up the Cluster Infrastructure on AWS EC2

This section describes step by step guide to setup an AWS account and launch the AWS EC2 free tier eligible Ubuntu servers. These servers will be used to setup a four node Apache Hadoop Clusters on AWS EC2 cloud infrastructure.

Section 2: Installing Apache Hadoop and Setting up the Cluster

This section provides step by step guide to install pre-requisites for Hadoop Installation and to configure the cluster on EC2 servers. The section explains primary Hadoop configuration files, Password-less SSH access, configuring master and slaves, and service start/stop in detail.

Note
The configuration described here is intended for learning purposes only.

1. Installing Apache Hadoop and setting up the Cluster

After creating and configuring cluster servers at AWS EC2, the AWS EC2 infrastructure is now ready to start installation and configuration of Apache Hadoop Cluster. This section describes the steps in details to install Apache Hadoop and configure a 4-Node Apache Hadoop cluster.

1.1 Prepare the servers for Hadoop installation

1.1.1. Update the packages and their dependencies

The first task is to run **'apt-get update'** to download the package lists from the repositories and "update" them to get information on the newest versions of packages and their dependencies.

```
$sudo apt-get update
```

FIGURE 1-1. UPDATE THE PACKAGES

```
ubuntu@ip-10-251-81-223:~$ sudo apt-get update
Get:1 http://us-west-2.ec2.archive.ubuntu.com precise Release.gpg [198 B]
Get:2 http://us-west-2.ec2.archive.ubuntu.com precise-updates Release.gpg [198 B
]
Get:3 http://us-west-2.ec2.archive.ubuntu.com precise Release [49.6 kB]
```

1.1.2 Install the Java for Hadoop 1.2.0

Use apt-get to install the JDK 6 on the server.

```
$sudo apt-get install openjdk-6-jdk
```

FIGURE 1-2 INSTALL JDK

```
ubuntu@ip-10-251-81-223:~$ sudo apt-get install openjdk-6-jdk
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  ca-certificates-java fontconfig fontconfig-config hicolor-icon-theme
  icedtea-6-jre-cacao icedtea-6-jre-jamvm icedtea-netx icedtea-netx-common
```

1.1.3 Download the Hadoop package

Download the binaries to your home directory. Use the default user 'ubuntu' for the installation.

In Live production instances a dedicated Hadoop user account for running Hadoop is used. Though, it's not mandatory to use a dedicated Hadoop user account but is recommended because this helps to separate the Hadoop installation from other software applications and user accounts running on the same machine (separating for security, permissions, backups, etc.).

```
$wget http://archive.apache.org/dist/hadoop/core/hadoop-1.2.0/hadoop-1.2.0.tar.gz
```

FIGURE 1-3. DOWNLOAD HADOOP 1.2.0

```
ubuntu@ip-10-251-81-223:~$
ubuntu@ip-10-251-81-223:~$ wget http://archive.apache.org/dist/hadoop/core/hadoop-1
.2.0/hadoop-1.2.0.tar.gz
--2013-09-21 22:20:31--  http://archive.apache.org/dist/hadoop/core/hadoop-1.2.0/ha
doop-1.2.0.tar.gz
Resolving archive.apache.org (archive.apache.org)... 140.211.11.131, 192.87.106.229
, 2001:610:1:80bc:192:87:106:229
Connecting to archive.apache.org (archive.apache.org)|140.211.11.131|:80... connect
ed.
HTTP request sent, awaiting response... 200 OK
Length: 62984953 (60M) [application/x-gzip]
Saving to: `hadoop-1.2.0.tar.gz'

3% [>                                        ] 2,232,974   5.31M/s
```

Unzip the files and review the package content and configuration files.

```
$tar -xvf hadoop-1.2.0.tar.gz
```

FIGURE 1-4. HADOOP PACKAGE CONTENT

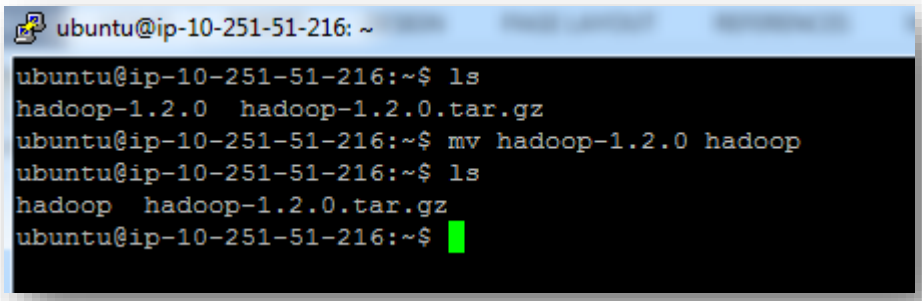
```
ubuntu@ip-10-251-81-223:~$ ls
hadoop-1.2.0  hadoop-1.2.0.tar.gz  keypairs
ubuntu@ip-10-251-81-223:~$ unalias -a
ubuntu@ip-10-251-81-223:~$ cd
ubuntu@ip-10-251-81-223:~$ ls
hadoop-1.2.0  hadoop-1.2.0.tar.gz  keypairs
ubuntu@ip-10-251-81-223:~$ cd hadoop-1.2.0/
ubuntu@ip-10-251-81-223:~/hadoop-1.2.0$ ls
bin          hadoop-ant-1.2.0.jar          ivy          sbin
build.xml    hadoop-client-1.2.0.jar       ivy.xml      share
c++          hadoop-core-1.2.0.jar         lib          src
CHANGES.txt  hadoop-examples-1.2.0.jar     libexec      webapps
conf         hadoop-minicluster-1.2.0.jar  LICENSE.txt
contrib      hadoop-test-1.2.0.jar         NOTICE.txt
docs         hadoop-tools-1.2.0.jar        README.txt
ubuntu@ip-10-251-81-223:~/hadoop-1.2.0$ cd conf/
ubuntu@ip-10-251-81-223:~/hadoop-1.2.0/conf$ ls
capacity-scheduler.xml      hadoop-policy.xml      slaves
configuration.xsl           hdfs-site.xml          ssl-client.xml.example
core-site.xml               log4j.properties       ssl-server.xml.example
fair-scheduler.xml          mapred-queue-acls.xml  taskcontroller.cfg
hadoop-env.sh               mapred-site.xml        task-log4j.properties
hadoop-metrics2.properties  masters
ubuntu@ip-10-251-81-223:~/hadoop-1.2.0/conf$
```

Review the Hadoop configurations files.

1.1.4 Update ".bashrc" file for user 'ubuntu'.

Rename the 'hadoop-1.2.0' directory to 'hadoop' for ease of operation and maintenance.

FIGURE 1-5 MOVE HADOOP1-2.0 TO HADOOP



Update the .bashrc file to add important Hadoop paths and directories.

a) Change directory to home.

```
$ cd
```

b) Edit the file

```
$ vi .bashrc
```


----------Set Hadoop environment Variables - Begin----------

```
# Set Hadoop-related environment variables
export CONF=/home/ubuntu/hadoop/conf

# Set JAVA_HOME (we will also configure JAVA_HOME for Hadoop
execution later on)

export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-amd64

# Add Hadoop bin/ directory to PATH

export PATH=$PATH:$/home/ubuntu/hadoop/bin
```

----------Set Hadoop environment Variables – End ----------

FIGURE 1-6 EDIT .BASHRC

```
# Set Hadoop-related environment variables

export CONF=/home/ubuntu/hadoop/conf

# Set JAVA_HOME (we will also configure JAVA_HOME directly for Hadoop later on)
export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-amd64

# Add Hadoop bin/ directory to PATH
export PATH=$PATH:$/home/ubuntu/hadoop/bin

# If not running interactively, don't do anything
```

c) Source the .bashrc file to set the hadoop environment variables without having to invoke a new shell:

```
$. ~/.bashrc
```

Execute the all the steps of this section on all the remaining cluster servers.

1.2 Setup Password-less SSH on Servers

Master server remotely starts services on salve nodes. This requires password-less access to Slave Servers. Ubuntu server at AWS comes with a pre-installed OpenSSh server.

1.2.1 Add the AWS EC2 Key Pair identity to SSH profile

The **'ssh-agent'** is a background program that handles passwords for SSH private keys.

The **'ssh-add'** command prompts the user for a private key password and adds it to the list maintained by ssh-agent. Once you add a password to ssh-agent, you will not be asked to provide the key when using SSH or SCP to connect to hosts with your public key.

Note

The public part of the key loaded into the agent must be put on the target system in **~/.ssh/authorized_keys**. This has been taken care of by the AWS Server creation process Step 1.3 Creating Cluster member servers

Because AWS EC2 server creation has taken care of **'authorized_keys',** on Master server, execute following steps to enable the password-less access to slave servers.

a) Protect key files to avoid any accidental or intentional corruption.

```
$chmod 644 authorized_keys

$chmod 400 edurekahadoop4ncluster.pem
```

b) Start ssh-agent

```
$eval `ssh-agent -s`
```

c) Add the secure identity to SSH Agent Key repository

```
$ ssh-add edurekahadoop4ncluster.pem
```

FIGURE 1-7 PROTECT THE KEYS AND UPDATE THE SSH AGENT REPOSITORY

```
ubuntu@ip-10-251-46-75: ~/.ssh
ubuntu@ip-10-251-46-75:~$ pwd
/home/ubuntu
ubuntu@ip-10-251-46-75:~$ ls
edurekahadoop4ncluster.pem
ubuntu@ip-10-251-46-75:~$ cp edurekahadoop4ncluster.pem .ssh/
ubuntu@ip-10-251-46-75:~$ cd .ssh/
ubuntu@ip-10-251-46-75:~/.ssh$ ls
authorized_keys  edurekahadoop4ncluster.pem
ubuntu@ip-10-251-46-75:~/.ssh$ chmod 644 authorized_keys
ubuntu@ip-10-251-46-75:~/.ssh$ chmod 400 edurekahadoop4ncluster.pem
ubuntu@ip-10-251-46-75:~/.ssh$ eval `ssh-agent -s`
Agent pid 1240
ubuntu@ip-10-251-46-75:~/.ssh$ ssh-add edurekahadoop4ncluster.pem
Identity added: edurekahadoop4ncluster.pem (edurekahadoop4ncluster.pem)
ubuntu@ip-10-251-46-75:~/.ssh$
```

If you don't protect the server files in 2.2.1 Add the AWS EC2 Key Pair identity to SSH profile, you would get following error in running 'ssh-add':

FIGURE 1-8 ERROR IN ADDING THE IDENTITY TO SSH AGENT REPOSITORY

```
ubuntu@ip-10-251-81-223:~/.ssh$ ssh-add hadoopcluster5n.pem
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0664 for 'hadoopcluster5n.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
ubuntu@ip-10-251-81-223:~/.ssh$
```

1.2.2 Test the password-less SSH access

Test the password-less SSH access by accessing (SSH) one of the slave node.

FIGURE 1-9 TEST THE PASSWORD-LESS ACCESS

```
ubuntu@ec2-54-218-170-127: ~
ubuntu@ip-10-251-46-75:~/.ssh$ ssh ubuntu@ec2-54-218-170-127.us-west-2.compute.amazonaws
.com
The authenticity of host 'ec2-54-218-170-127.us-west-2.compute.amazonaws.com (10.251.46.
141)' can't be established.
ECDSA key fingerprint is e2:30:e7:16:db:43:02:20:6f:9e:38:6c:84:53:6b:c7.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-54-218-170-127.us-west-2.compute.amazonaws.com,10.251.46
.141' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.2.0-40-virtual x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Thu Sep 26 20:32:03 UTC 2013

  System load:  0.0               Processes:           60
  Usage of /:   11.0% of 7.87GB   Users logged in:     1
  Memory usage: 8%                IP address for eth0: 10.251.46.141
  Swap usage:   0%

  Graph this data and manage this system at https://landscape.canonical.com/

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

  Use Juju to deploy your cloud instances and workloads:
    https://juju.ubuntu.com/#cloud-precise

0 packages can be updated.
0 updates are security updates.

Last login: Thu Sep 26 20:18:21 2013 from 182.72.99.30
ubuntu@ec2-54-218-170-127:~$
```

Test the password-less access from master server to all the other servers in cluster. Notice the IP address difference.

1.3 Setup the Hadoop Cluster

This section describes the detail steps needed for setting up the Hadoop Cluster and configuring the core Hadoop configuration files.

1.3.1 Configure JAVA_HOME

Configure JAVA_HOME in **'hadoop-env.sh'**. This file specifies environment variables that affect the JDK used by Hadoop Daemons started by the Hadoop start-up scripts:

```
$cd $CONF
```

```
$vi hadoop-env.sh
```

Remove the comment and update the JAVA_HOME to:

```
export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-amd64
```

FIGURE 1-10 JAVA HOME SETUP

```
# The java implementation to use.  Required.
export JAVA_HOME=/usr/lib/jvm/java-1.6.0-openjdk-amd64
```

1.3.2. Configure the Default Filesystem

This file contains the configuration settings for Hadoop Core such as I/O settings that are common to HDFS and MapReduce. Configure default files-system (Parameter: fs.default.name) used by clients in **core-site.xml**

```
$cd $CONF
```

```
$vi core-site.xml
```

Add the following line in between the configuration tag:

<configuration>

<property>

<name>fs.default.name</name>

<value>hdfs://ec2-54-214-206-65.us-west-2.compute.amazonaws.com:8020</value>

</property>

</configuration>

FIGURE 1-11 CONFIGURE THE DEFAULT FILE SYSTEM

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://ec2-54-214-206-65.us-west-2.compute.amazonaws.com:8020</value>
</property>
</configuration>
```

Where *hostname* and *port* are the machine and port on which Name Node daemon runs and listens. It also informs the Name Node as to which IP and port it should bind. The commonly used port is 8020 and you can also specify IP address rather than hostname.

Note
For the simplicity of understanding the cluster setup, we have changed only necessary parameters to start a cluster.

1.3.3 Configure the HDFS

This file contains the cconfiguration settings for HDFS daemons; the Name Node, the secondary Name Node, and the data nodes.

Configure **hdfs-site.xml** and specify default block replication and permission checking on HDFS. The actual number of replications can be specified when the file is created. The default is used if replication is not specified in create time. If "true", enable permission checking in HDFS. If "false", permission checking is turned off, but all other behaviour is unchanged. Switching from one parameter value to the other does not change the mode, owner or group of files or directories.

```
$cd $CONF

$vi hdfs-site.xml
```

Add the following line in between the configuration tag:

<configuration>

<property>

<name>dfs.replication</name>

<value>2</value>

</property>

<property>

<name>dfs.permissions</name>

<value>false</value>

</property>

</configuration>

FIGURE 1-12 CONFIGURE THE DEFAULT FILESYSTEM

```
ubuntu@ip-10-251-81-223: ~/hadoop-1.2.0/conf
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
<name>dfs.replication</name>
<value>3</value>
</property>
<property>
<name>dfs.permissions</name>
<value>false</value>
</property>

</configuration>
```

1.3.4 Configure the Job Tracker

This file contains the configuration settings for MapReduce daemons; the job tracker and the task-trackers.

Configure **mapred-site.xml** and specify Job Tracker details. The **mapred.job.tracker** parameter is a *hostname* (or IP address) and *port* pair on which the Job Tracker listens for RPC communication. This parameter specify the location of the Job Tracker to Task Trackers and MapReduce clients.

```
$cd $CONF

$vi mapred-site.xml
```

Add the following line in between the configuration tag:

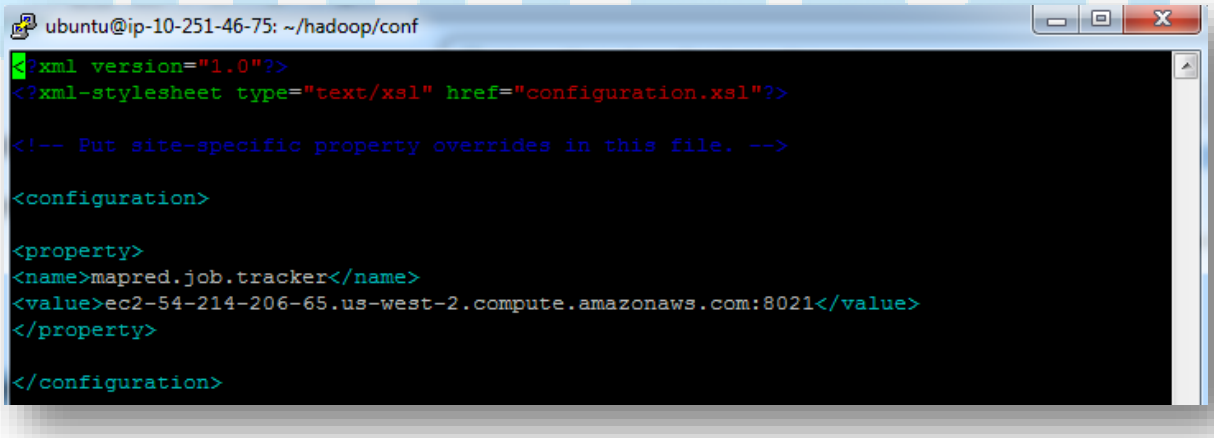
<property>

<name>mapred.job.tracker</name>

<value> hdfs://ec2-54-214-206-65.us-west-2.compute.amazonaws.com:8021</value>

</property>

FIGURE 1-13 CONFIGURE THE JOBTRACKER DETAILS



The master server configuration is completed and you can move these files to slaves.

1.3.5 Move the configuration files to Slave servers

Post configuration, copy these core configuration files to all the slave servers using SCP (Secure copy).

FIGURE 1-14 COPYING THE FILES TO SLAVE SERVERS

```
ubuntu@ip-10-251-46-75: ~/hadoop/conf
ubuntu@ip-10-251-46-75:~/hadoop/conf$ scp hadoop-env.sh core-site.xml hdfs-site.xml mapr
ed-site.xml ubuntu@ec2-50-112-231-12.us-west-2.compute.amazonaws.com:/home/ubuntu/hadoop
/conf
The authenticity of host 'ec2-50-112-231-12.us-west-2.compute.amazonaws.com (10.251.51.2
16)' can't be established.
ECDSA key fingerprint is 04:ff:55:ca:6b:88:ad:ff:82:92:63:4a:f9:7b:9d:84.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-50-112-231-12.us-west-2.compute.amazonaws.com,10.251.51.
216' (ECDSA) to the list of known hosts.
hadoop-env.sh                                     100% 2446     2.4KB/s   00:00
core-site.xml                                     100%  306     0.3KB/s   00:00
hdfs-site.xml                                     100%  319     0.3KB/s   00:00
mapred-site.xml                                   100%  304     0.3KB/s   00:00
ubuntu@ip-10-251-46-75:~/hadoop/conf$ scp hadoop-env.sh core-site.xml hdfs-site.xml mapr
ed-site.xml ubuntu@ec2-54-218-170-127.us-west-2.compute.amazonaws.com:/home/ubuntu/hadoo
p/conf
hadoop-env.sh                                     100% 2446     2.4KB/s   00:00
core-site.xml                                     100%  306     0.3KB/s   00:00
hdfs-site.xml                                     100%  319     0.3KB/s   00:00
mapred-site.xml                                   100%  304     0.3KB/s   00:00
ubuntu@ip-10-251-46-75:~/hadoop/conf$ scp hadoop-env.sh core-site.xml hdfs-site.xml mapr
ed-site.xml ubuntu@ec2-54-202-24-115.us-west-2.compute.amazonaws.com:/home/ubuntu/hadoop
/conf
The authenticity of host 'ec2-54-202-24-115.us-west-2.compute.amazonaws.com (10.251.89.2
21)' can't be established.
ECDSA key fingerprint is 4e:ee:13:d4:bf:2c:ea:30:47:0c:18:e1:5a:2f:71:02.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-54-202-24-115.us-west-2.compute.amazonaws.com,10.251.89.
221' (ECDSA) to the list of known hosts.
hadoop-env.sh                                     100% 2446     2.4KB/s   00:00
core-site.xml                                     100%  306     0.3KB/s   00:00
hdfs-site.xml                                     100%  319     0.3KB/s   00:00
mapred-site.xml                                   100%  304     0.3KB/s   00:00
ubuntu@ip-10-251-46-75:~/hadoop/conf$
```

1.3.6 Configure Master and Slaves

Edit the master and slave files to configure the masters and slaves in the Hadoop Cluster.

1.3.6.1 Configure Master Server

The 'masters' file at Master server contains a list of hosts, one per line, that are to host Secondary Name Node servers. Update 'master' file at **Master Node** using VI Editor.

The 'slaves' file at Master node contains a list of hosts, one per line, that are to host Data Node and Task Tracker servers. Update 'slave' file at Master node using VI Editor.

FIGURE 1-15 CONFIGURE 'SLAVES' FILE ON MASTER SERVER

```
ubuntu@ip-10-251-46-75: ~/hadoop/conf
ec2-54-218-170-127.us-west-2.compute.amazonaws.com
ec2-54-202-24-115.us-west-2.compute.amazonaws.com
```

Move (Using SCP) the 'masters' and 'salves' files from Name Node to Secondary Name Node as it has the same configuration as Name Node.

Note

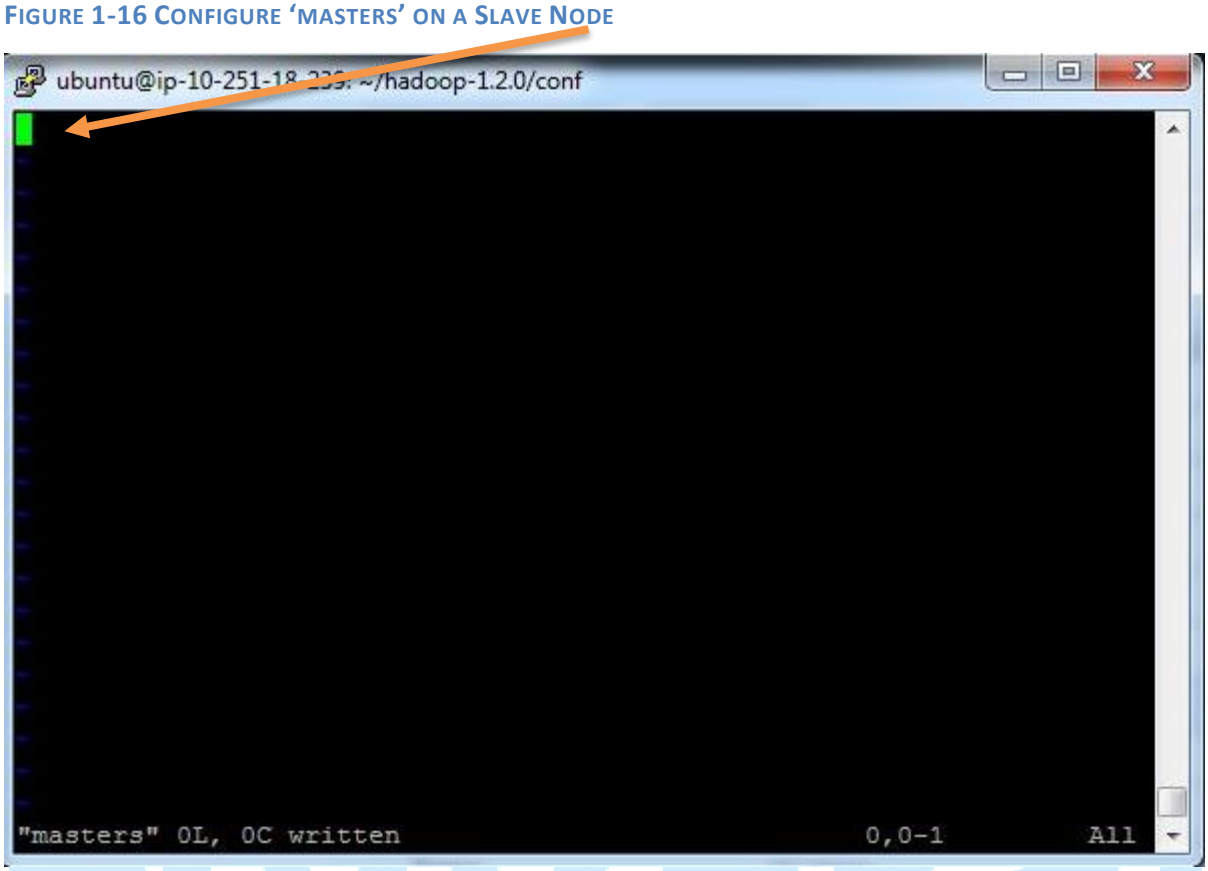
If you are building a test cluster, you don't need to set up secondary name node on a different machine, something like pseudo cluster install steps. However if you're building out a real distributed cluster, you must move secondary node to other machine.

We are setting up a Secondary Name Node on a different machine similar to a real Hadoop distributed cluster.

1.3.6.2 Configure Slave servers

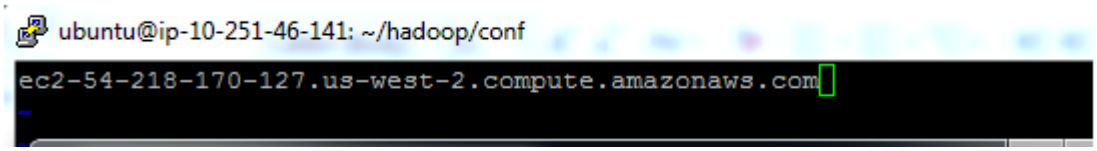
Update the 'masters' files on **Slave Node.** Notice that the file is blank to configure the slave node.

FIGURE 1-16 CONFIGURE 'MASTERS' ON A SLAVE NODE



Update the 'slaves' file on Slave server with the IP address of the slave node. Notice that the 'slaves' file at Slave node contains only its own IP address and not of any other Data Node in the cluster.

FIGURE 1-17 CONFIGURE 'SLAVES' ON A SLAVE NODE



Similarly, update the other slave nodes.

1.3.7 Start the DFS services

The first step in starting up your Hadoop installation is formatting the Hadoop file-system, which is implemented on top of the local file-systems of your cluster. This is required on the first time

Hadoop installation. Do not format a running Hadoop file-system, this will cause all your data to be erased.

To format the file-system, run the command:

```
$hadoop namenode -format
```

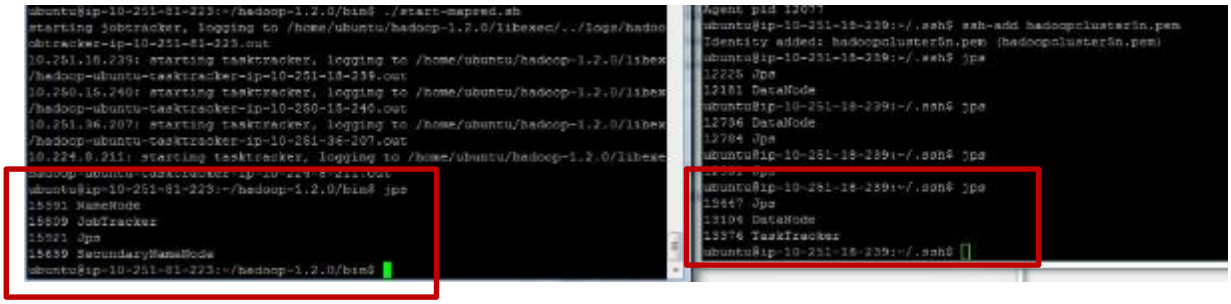
You are now all set to start the HDFS services i.e. Name Node, Secondary Name Node, and Data Nodes on your Apache Hadoop Cluster.

FIGURE 1-18 START THE SERVICES

```
ubuntu@ip-10-251-81-223:~/hadoop-1.2.0/bin$ ./start-dfs.sh
starting namenode, logging to /home/ubuntu/hadoop-1.2.0/libexec/../logs/hadoop-ubuntu-nam
enode-ip-10-251-81-223.out
10.224.8.211: starting datanode, logging to /home/ubuntu/hadoop-1.2.0/libexec/../logs/had
oop-ubuntu-datanode-ip-10-224-8-211.out
10.251.18.239: starting datanode, logging to /home/ubuntu/hadoop-1.2.0/libexec/../logs/ha
doop-ubuntu-datanode-ip-10-251-18-239.out
10.250.15.240: starting datanode, logging to /home/ubuntu/hadoop-1.2.0/libexec/../logs/ha
doop-ubuntu-datanode-ip-10-250-15-240.out
10.251.36.207: starting datanode, logging to /home/ubuntu/hadoop-1.2.0/libexec/../logs/ha
doop-ubuntu-datanode-ip-10-251-36-207.out
10.251.81.223: starting secondarynamenode, logging to /home/ubuntu/hadoop-1.2.0/libexec/.
./logs/hadoop-ubuntu-secondarynamenode-ip-10-251-81-223.out
ubuntu@ip-10-251-81-223:~/hadoop-1.2.0/bin$
```


Start the Map Reduce services i.e. JobTracker and TaskTrackers and cross the service start-up using JPS (Java Process Monitoring Tool).

FIGURE 1-19 START THE MAPREDUCE SERVICES

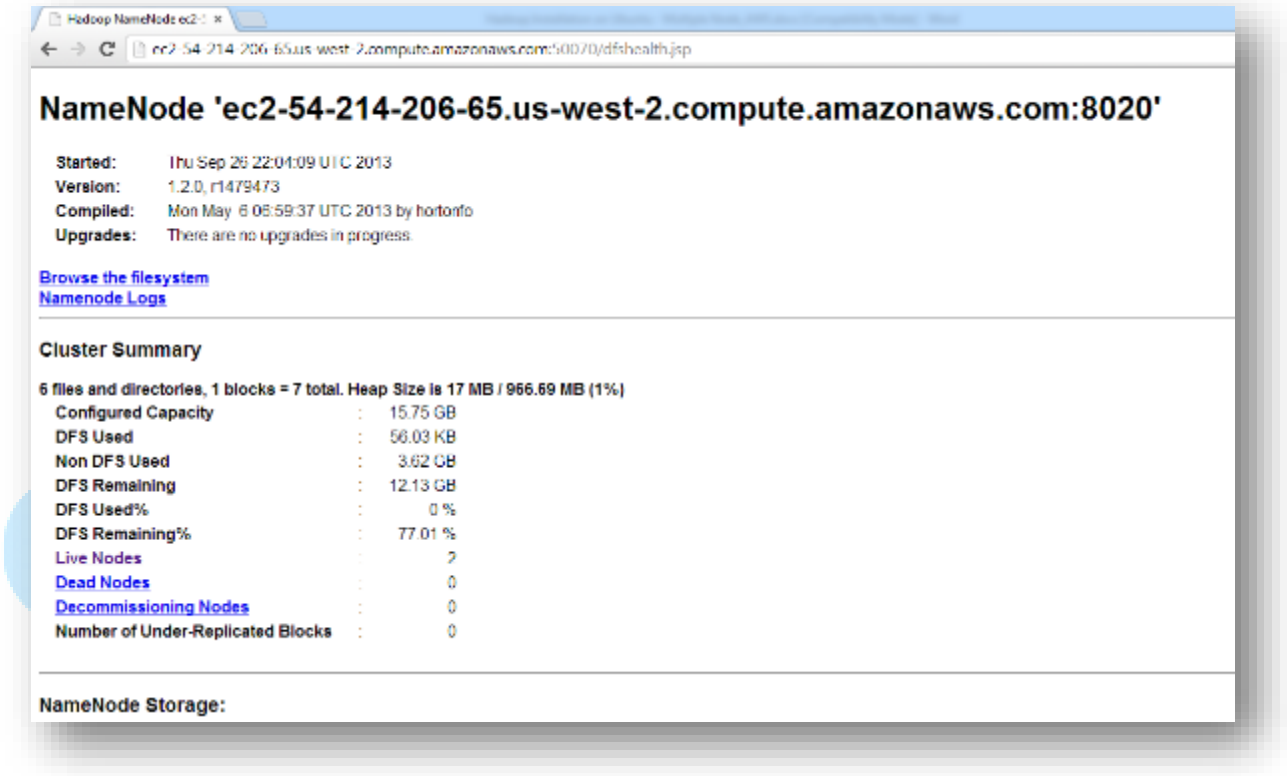


1.3.8 Perform the Health Check

a) Check the NameNode status:

http://ec2-54-214-206-65.us-west-2.compute.amazonaws.com:50070/dfshealth.jsp

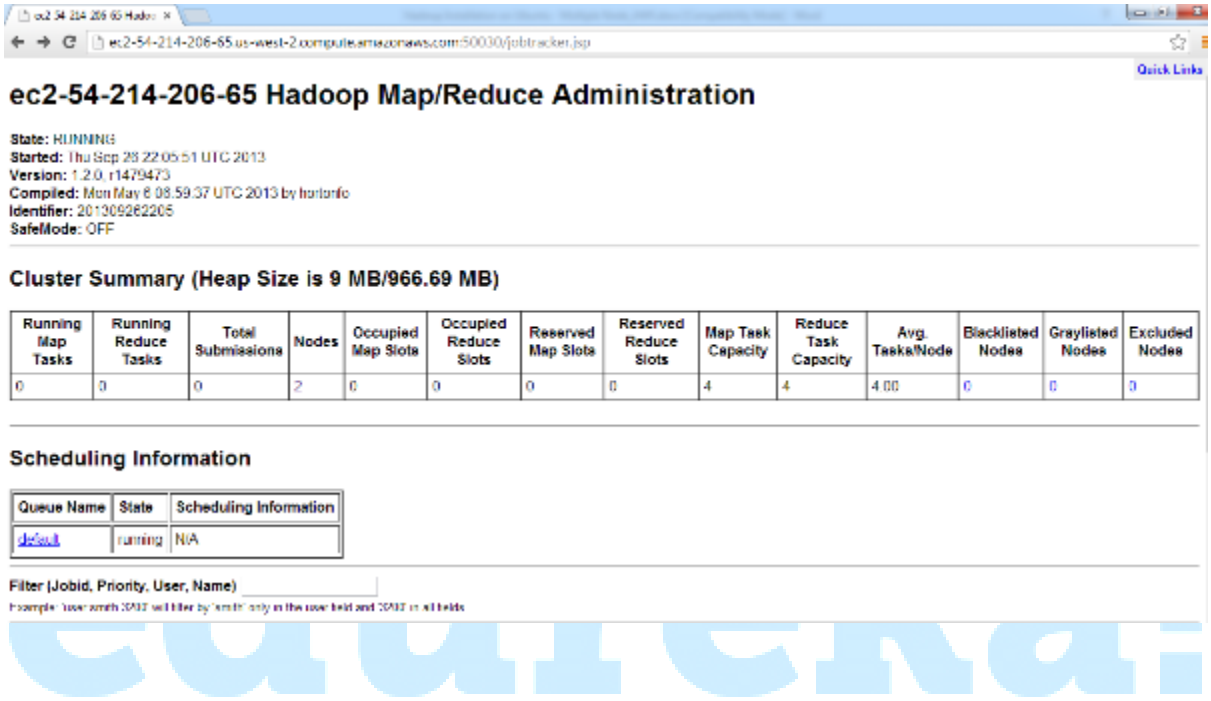
FIGURE 1-20 NAMENODE STATUS



b) JobTracker status:

http://ec2-54-214-206-65.us-west-2.compute.amazonaws.com:50030/jobtracker.jsp

FIGURE 1-21 JOBTRACKER STATUS



c) TaskTracker status:

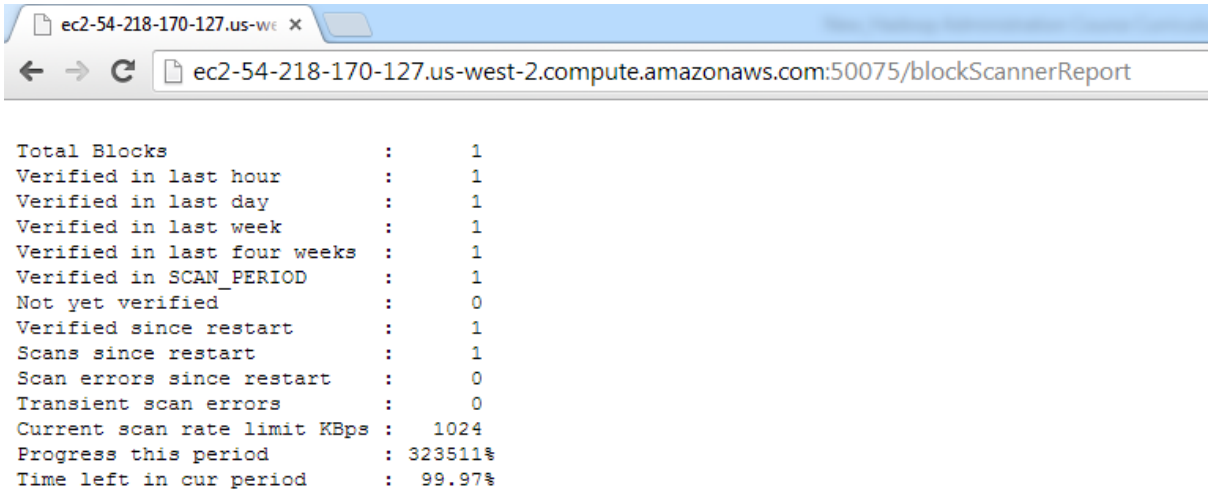
http://ec2-54-218-170-127.us-west-2.compute.amazonaws.com:50060/tasktracker.jsp

FIGURE 1-22 TASKTRACKER STATUS

d) Data Block Scanner Report:

http://ec2-54-218-170-127.us-west-2.compute.amazonaws.com:50075/blockScannerReport

FIGURE 1-23 DATABLOCK SCANNER REPORT



1.3.9 Terminate your AWS EC2 instances.

You can play around with the cluster, execute few Linux and HDFS hands-on commands, and submit a Job. Check the task status from the front end URL's.

Note

Remember to terminate or stop your instances after playing around with your Apache Hadoop Cluster at AWS. Idle stopped instances may attract additional charges if you have used any additional resource such as Elastic Block Storage, Elastic IP's etc. Review following AWS documentation for Start/Stop behavior: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Stop_Start.html

As explained in the AWS document, the Public URL and IP address will not be retained if you restart a stopped instance. Hence your saved putty or WINSCP sessions will not work. But the Key Pair is valid and can be used again with new Public URL.

You will lose all the configuration settings and software installations in case of termination.

FIGURE 1-24 TERMINATE YOUR AWS EC2 INSTANCES